43rd AIAA Aerospace Sciences Meeting and Exhibit
10-13 January 2005, Reno, Nevada

AIAA 2005-1369

SmaggIce: Further Progress in Software for Gridding 2D Iced Airfoils

Mary B. Vickerman*, Yung K. Choo†, Herbert W. Schilling‡, Donald C. Braun§, Marivell Baez**, Barbara J. Cotton††
NASA Glenn Research Center, Cleveland, OH, 44135

This paper presents progress being made in developing the SmaggIce software toolkit which is used to create structured grids for 2D iced airfoils in preparation for Computational Fluid Dynamics (CFD) analysis. A brief introduction is given which establishes the reasons for performing CFD analysis in icing research and the need for software to help with this. A short overview of a previously released version of SmaggIce (v1.2) is included. SmaggIce v1.2 includes tools for geometry preparation, a prerequisite task to performing gridding operations. Details of features added in the current version of SmaggIce (v1.8) are presented. These include tools for domain decomposition, block boundary modification, gridding, and grid quality display. Finally, plans are listed for the final version of SmaggIce (v2.0) which will include additional boundary modification tools and solution display.

I. Introduction

A. The need for CFD with 2D iced airfoils

Two dimensional icing aero analysis using CFD tools has its limitations and is not intended to address 3D issues. However, it is a cost-effective engineering tool that provides insights into the icing aerodynamics and preliminary performance results valuable to pre-tunnel and pre-flight test planning, as well as post-test analysis. The 2D CFD results need to be examined for grid independence, checked for their reasonableness to the physics of the problem, and interpreted correctly considering the inadequacies of turbulence models and the numerical method.[1]

Fully 3D steady/unsteady flow analysis over an aircraft (or even a finite wing) with highly 3D ice is very expensive in terms of human and computer time. In addition, validity of the numerical simulation will be challenged since there is no universally applicable turbulence model for separated flows with ice. On the other hand, direct numerical simulation (DNS), which doesn't require turbulence models, is too expensive to be a practical engineering tool even with today's massively parallel computers and will remain so for any foreseeable future.

Acquisition of 3D ice shape geometry is still very difficult and expensive.[2] Also, generating quality grids and simulating steady flow over 3D ice are both expensive and challenging tasks at the present time.[3] Unsteady flow simulation, even simplified detached eddy simulation (DES) with ice, will be much more expensive compared with the steady flow solutions of Reynolds Averaged Navier-Stokes (RANS) equations with parallel computing power for even a square wing section.[4]

Two dimensional CFD can support quasi-3D analysis, which has the potential to be a cost-effective engineering approach for 3D icing aero simulations. However, we have yet to see the development or effectiveness of a quasi-3D method that combines 2D CFD airfoil analysis[5] with lifting-line theory.[6,7,8]

B. Choice of structured grids

Flow-field characteristics over iced airfoils are affected by ice shapes in addition to other factors such as the angle of attack and freestream conditions. Therefore, accurate calculation of the aerodynamic performance of iced airfoils requires accurate calculation of the flow field around the ice features, which in turn requires having a high

* Computer Scientist, Computational Sciences Branch, 21000 Brookpark Rd./M.S. 142-5.
† Aerospace Engineer, Icing Branch, 21000 Brookpark Rd./M.S. 11-2, Member AIAA.
‡ Computer Scientist, Computational Sciences Branch, 21000 Brookpark Rd./M.S. 142-4.
§ Computer Engineer, Experimental Data Software Branch, 21000 Brookpark Rd./M.S. 142-5.
** Computer Scientist, Experimental Data Software Branch, 21000 Brookpark Rd./M.S. 142-5.
†† Computer Specialist, Experimental Data Software Branch, 21000 Brookpark Rd./M.S. 142-1.

quality grid for the flow domain to be simulated. Using blocked structured grids allows for more precise control of the grid structure and density in the flow field, compared to using unstructured grids. Although it is easier to automate the generation of unstructured grids, they generally require more cells and a corresponding increase in memory usage and CPU time when run through flow solvers. SmaggIce makes use of structured grids for the precise gridding control and better performance they provide in 2D. It overcomes the difficulties of creating these grids by providing partially automated blocking and gridding tools, yet also provides the necessary precise grid quality control through interactive boundary, block, and grid modification tools.[2, 9]

C. Gridding tool specifically for iced airfoils

When studying airfoil icing, a wide variety of ice shapes are encountered, all of which affect the flow field. To capture those ice-induced flow features, a grid generator has to provide effective grid creation and grid-quality controls. For icing aero analysis, handling of the ice-shape geometry cannot be separated from grid generation, since ice geometry imposes a particular challenge to grid generation.

SmaggIce is a 2D software toolkit that will streamline the simulation process for icing effects on airfoil performance, from geometry, to grid, to flow solution. The toolkit will provide tools to (1) take natural, simulated, or computer-generated ice shapes, and examine and modify them as needed,[10] (2) generate grids and evaluate and control their quality and density, (3) prepare files to use in running an application flow solver such as WIND[11], and (4) monitor the run. This procedure can be iterated to improve accuracy of the solution, as illustrated in Figure 1.

SmaggIce v1.8 provides tools to perform ice shape preparations, domain decomposition, block boundary modification, grid generation and modification, grid quality analysis, and grid output. SmaggIce v2.0 will additionally provide tools to perform additional grid modifications and to view CFD results. See Figure 1 for an overview of how SmaggIce fits into this process. With SmaggIce, icing aerodynamic analysis using CFD will become a practical engineering process that will provide insights to the flow phenomena and provide preliminary engineering answers on effects of ice on aerodynamic performance.

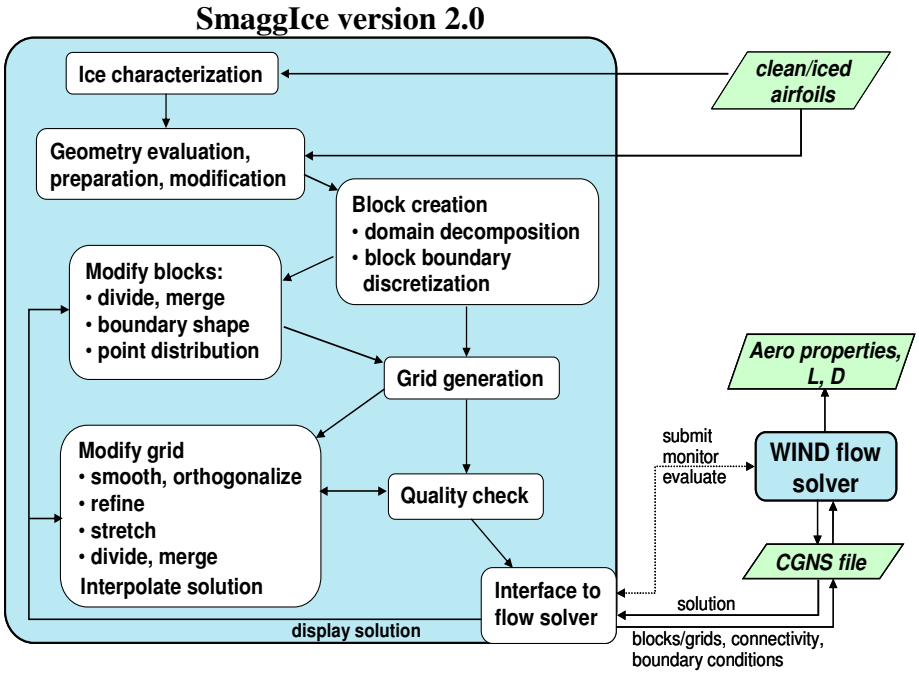

Figure 1. SmaggIce role in icing aerodynamic simulation and analysis process.

II. Summary of Version 1.2 Capabilities

Version 1.2 of SmaggIce for Unix operating systems was released in August 2002, and the Microsoft Windows version was released in March 2003 through the NASA Software Repository*. Its capabilities are detailed in Reference 12. We will present a brief summary of its capabilities here, as they relate to the domain decompositions and gridding aspects of the toolkit. Besides its capabilities to measure and record ice shape characteristics, SmaggIce allows the user to examine and prepare ice shape data for the grid generation process.

A. Ice shape characterization

Ice shape characterization tools are used to measure and record location, length, angle, arc length, and ice area. This gives users the means to measure the physical characteristics of ice such as icing limit locations, horn height

* https://technology.grc.nasa.gov/software

and angle, distance along the clean airfoil from the leading edge to a prominent ice location, or the area of ice between two user-specified points. These tools provide an aid in the process of determining the relationship between ice characteristics and their effects on aerodynamic performance.

B. Geometry preparation

In SmaggIce, an airfoil is defined as a sequence of points, i.e., (x,y) coordinates, starting at the upper (or lower) surface of the trailing edge and following along the surface toward the leading edge and then back toward the trailing edge along the lower (or upper) surface. Ice shapes are also defined by a sequence of points, although they do not have to include points back to the trailing edge. All modifications to the geometry are made to these points.

1. Boundary modification

Interactive ice shape control is used to prepare the 2D surface for gridding. Any subcurve (or the entire curve) of a surface can be selected for processing. The types of functions that can be applied to surfaces are smoothing, rediscretization, point redistribution, and correction of obvious input errors such as a twisted ice surface. Systematic smoothing of the iced surfaces in a controlled manner is accomplished using a control point formulation.[13] With this feature, a user can smooth irregular ice surfaces to a level acceptable to the user's grid generation tools. Users can control the level of smoothing by choosing the number of control points in constructing curves. Direct reshaping of the curve is done by dragging control points associated with the curve. Rediscretization provides a means of increasing/decreasing the number of points, distributing the points by curvature, and controlling the uniformity of their distribution. In addition, hyperbolic tangent stretching is provided. These control features of SmaggIce not only prepare the ice surface for gridding and CFD flow simulation, but they also allow users to correct any deficiencies (e.g., twists, gaps, too many or too few points) in the input data.

2. Create and place artificial ice

A tool is provided to attach artificial (i.e., computer-generated) ice shapes such as triangles, rectangles, quarter-circles, half-circles, and trapezoids to the surface of an airfoil. Parameters for location, replication, size, and number of points can be specified. This tool facilitates studies of the effects that various ice shapes and ice roughness have on aerodynamic performance.

III. Version 1.8 Capabilities

A. Ice shape preparation

Additional tools to prepare an ice shape for gridding are provided in the current version of SmaggIce. One of these tools can extend the trailing edge of an "open" airfoil (i.e., one whose first and last points do not coincide). Another can extend the geometry of an "open" ice shape (i.e., one whose geometry does not include the complete airfoil, but only the geometry for the ice accretion at the leading edge) to the clean airfoil. This is used to create iced airfoil geometry by combining clean airfoil data with open ice data from a tracing as reported in Reference 14.

B. Domain decomposition

Domain decomposition divides the flow domain into blocks, and is performed prior to gridding. The domain is decomposed in steps, with various types of blocks created during each step.

1. Wake definition

The wake extends downstream from the trailing edge of the iced airfoil. This is used to set up the cut for the C-shaped domain that will be used during domain decomposition. During wake definition, the user specifies the number of points in the wake, the length, and the angle of the wake. The spacing of the wake points closest to the trailing edge is based on the spacing of points on the airfoil at the trailing edge. The rest of the point spacings are set to accommodate the specified length and number of points. Wake definition in SmaggIce is shown in Figure 2.

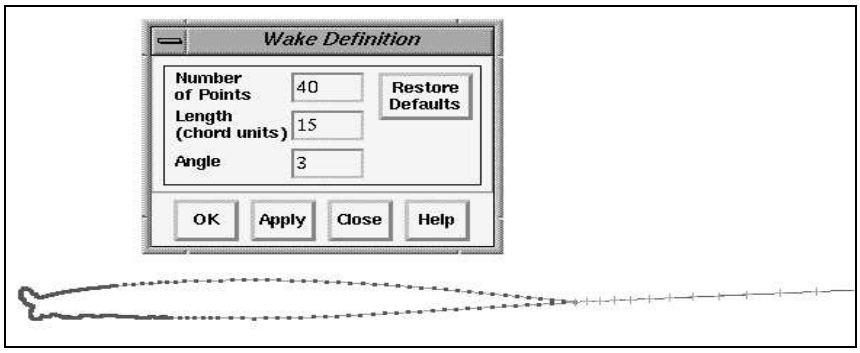

Figure 2. Creating a wake.

2. Viscous sublayer block

The viscous sublayer block is a thin C-shaped block that wraps around the wake and iced airfoil. This block (1) serves as a transition layer from rugged ice to a smoother outer boundary of this block, and (2) provides a very dense mesh near the no-slip boundary with firm control. The viscous sublayer is specified by the number of points in the radial direction, the thickness of the block, and the initial grid spacing (adjacent to the iced airfoil and wake). The spacing of the points in the radial direction is determined by these three parameters. The points along the inner boundary of the viscous sublayer block are created to be in the same locations as the points along the wake and iced airfoil. Creation of the viscous sublayer is shown in Figure 3.

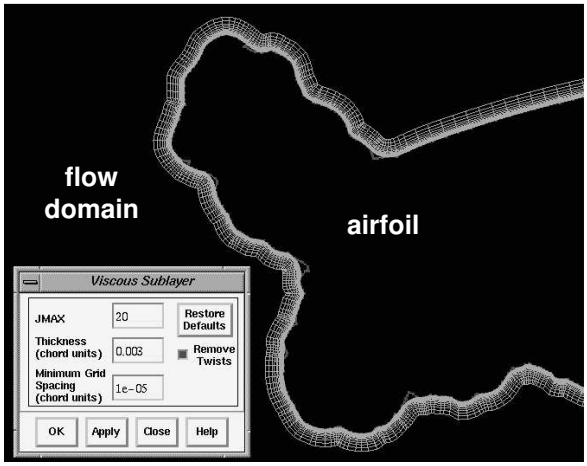

Figure 3. Viscous Sublayer

Concave areas of ice can introduce tangles in the viscous sublayer block. These tangles can be removed by reducing the thickness of the block, manually smoothing the ice surface before creating the block, or using an automatic process to smooth out the inner and outer edges of the viscous sublayer block. The effects of automatic smoothing can be seen in Figure 4.

In the case of airfoils with "horn-shaped ice" such as the one shown in Figure 3, whose flow is dominated by separated flows aft of the suction-side horn, a previous study indicates that the location of flow separation at the horn tip is critical to the airfoil performance, but detailed geometry of other parts of the ice is not critical.[15, 16] It should be noted that the upper horn-tip region on the suction side is convex and is not affected by the smoothing process.

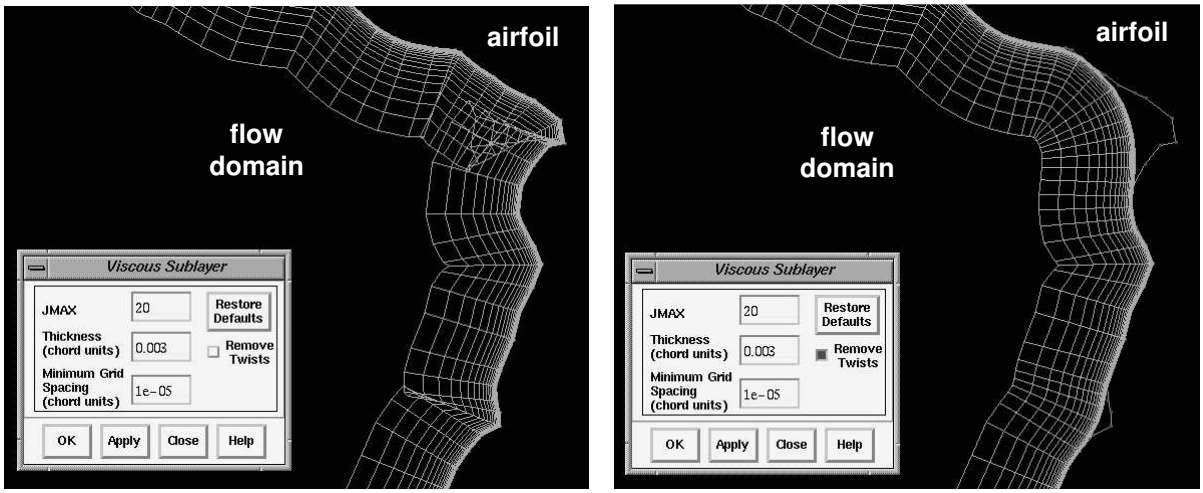

Figure 4. Viscous sublayer twisted in concave areas (left); smoothed (right).

3. Near field

Near field decomposition lets the user define the blocking scheme and parameters for the near field. The outer boundary of the near field consists of a semi-circle centered on the leading edge point and a parallelogram abutting the semi-circle and extending to the end of the wake. The inner boundary is created to be abutting one-to-one with the outer boundary of the viscous sublayer block. The initial grid cell spacing along that inner boundary is set to match the last cell spacing along the outer boundary of the viscous sublayer.

The user selects the basic blocking topology to be used when creating the near field. Current choices are *single block* (for clean airfoils or airfoils with non-complex ice shapes) and *radial cut* (for more complicated ice shapes). Parameters are specified to set the radius of the leading edge semi-circle, the number of points in the radial direction, and for radial cut topology, the number of radial cuts to use. For radial cut topology, the user can interactively select and move the endpoints of the radial cuts, then shape the cuts, as shown in Figure 5. When the near field domain decomposition is complete, the near field blocks are created and grids are generated for each of those blocks.

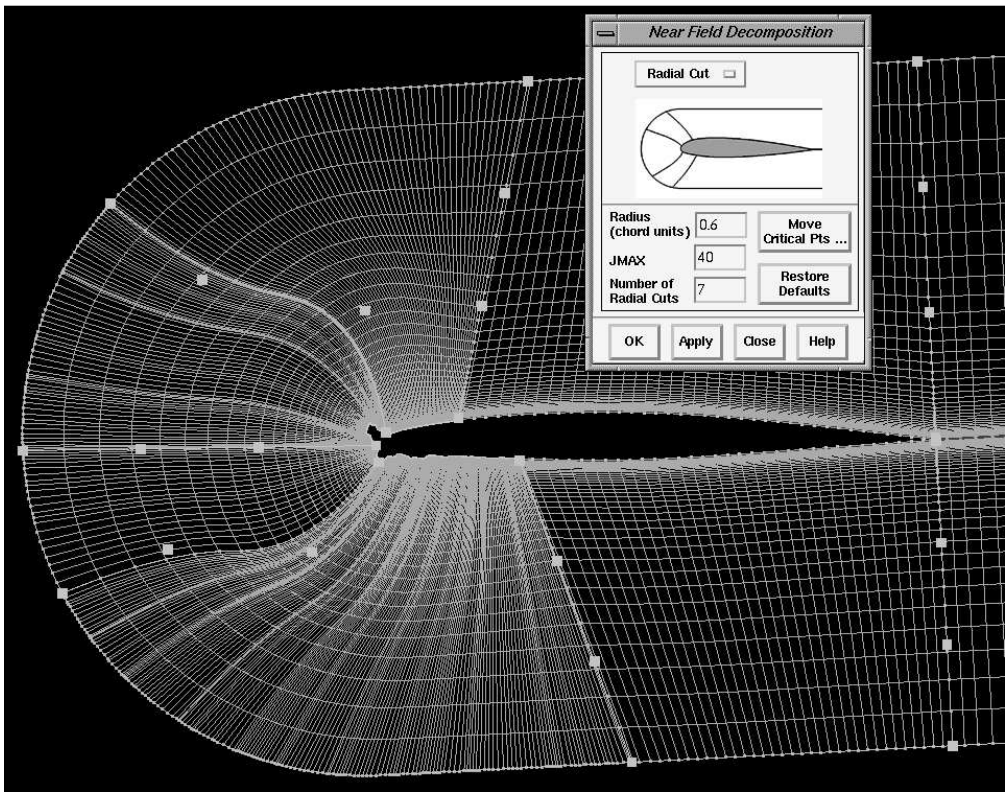

Figure 5. Near field with modifiable radial cuts.

4. Outer block

The user defines parameters for generating the outer block. The outer boundary of the outer block consists of a semi-circle centered on the leading edge point and a rectangle abutting the semi-circle and extending to the end of the wake, as shown in Figure 6 and Figure 7. The inner boundary is created to overlap the outer boundary of the near field. The user specifies the number of cells to overlap, the number of points in the I-direction and J-direction of the grid, and the outer radius of the semi-circle at the leading edge. After the parameters are specified, the boundaries of the outer block and its grid are generated.

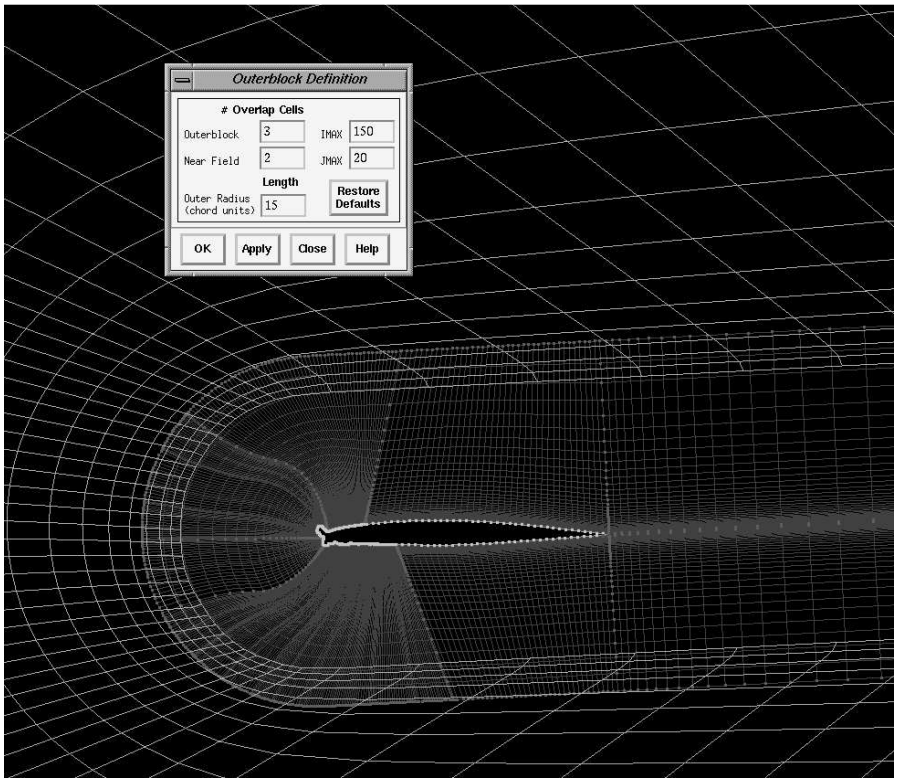

Figure 6. Outer block, close-up view.

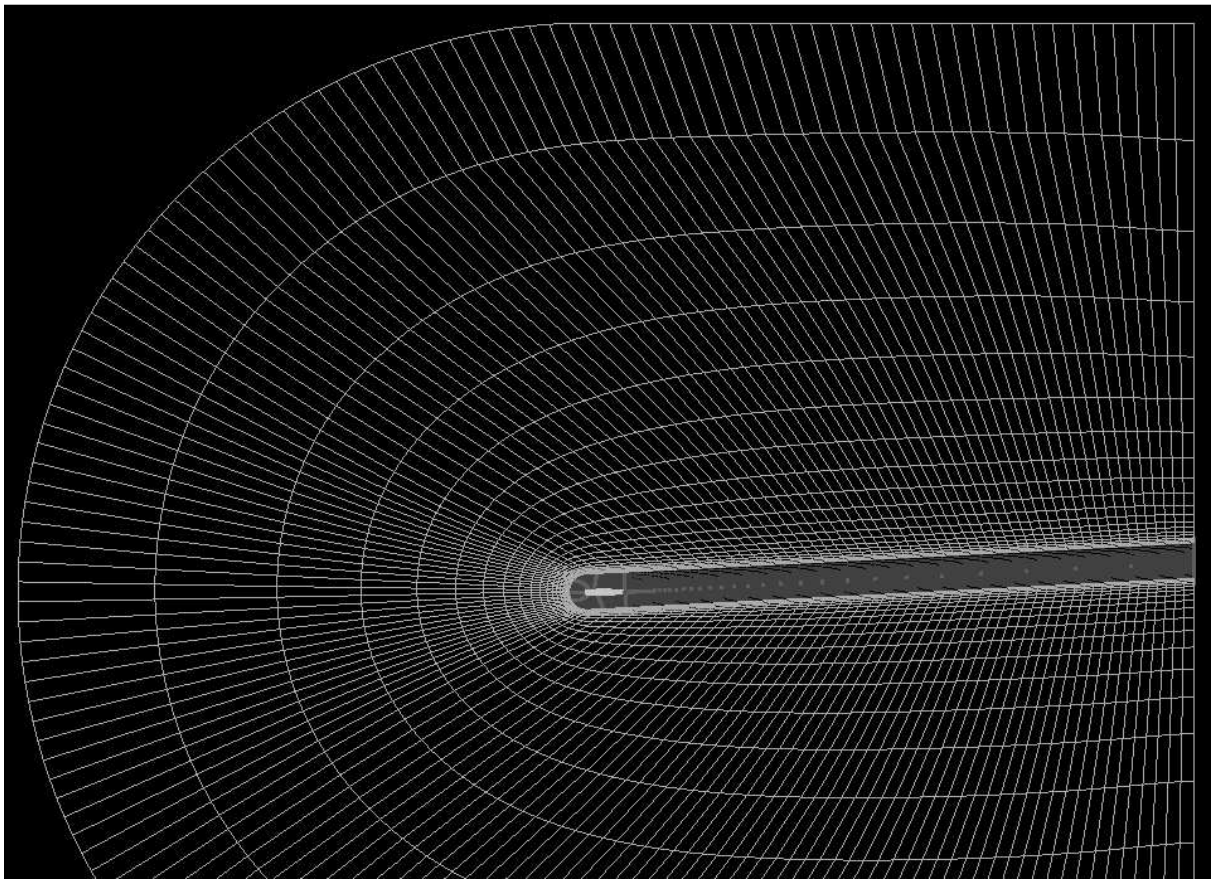

Figure 7. Outer block, wide view.

5. Boundary conditions

During domain decomposition, appropriate boundary conditions are assigned to block boundaries. These boundary conditions are stored along with the geometry for each block and output with the geometry when saving to a CGNS file. This saves the user the extra step of having to use a separate program to define the boundary conditions.

6. Block connectivity

The viscous sublayer block and near field blocks are created to have one-to-one abutting connectivity with each other, while the near field blocks and the outer block have overlapping (overset) connectivity between them. These connectivities are stored for each block, along with the geometry. When modifications are made to block boundaries (rediscretizing, redistributing points, dividing and merging grids), the connectivity information is maintained or modified as necessary. The connectivity information is output with the geometry when saving to a CGNS file. This saves the user the extra step of having to use a separate program to define the connectivities between blocks.

Inverse bilinear interpolation is used to compute the overlapping connectivities – at boundary points of the outermost near field blocks that overlap the outer block and at the boundary points of the outer block that overlap near field blocks. An efficient and robust algorithm was developed that uses fast algebraic calculations rather than an iterative procedure. The algorithm identifies and properly handles every possible degenerate special case of a quadrilateral cell.

7. Block boundary modifications

The same tools that are used for geometry preparation also can be used to modify block boundaries. For example, the Change Free Form tool can be used to adjust the shape of a block boundary after the block has been created. In step 1 (Figure 8), the subcurve of the boundary to be modified is selected. In step 2 (Figure 9), control points are dragged to change the shape of the subcurve. When the boundary is modified, the grid is removed, since it no longer matches the boundary. A preview of the adjusted points is shown. When the changes are applied, the abutting boundary from the adjacent block is also changed to keep the connectivity intact. In step 3 (Figure 10), the grids are regenerated for the two blocks whose boundaries were modified.

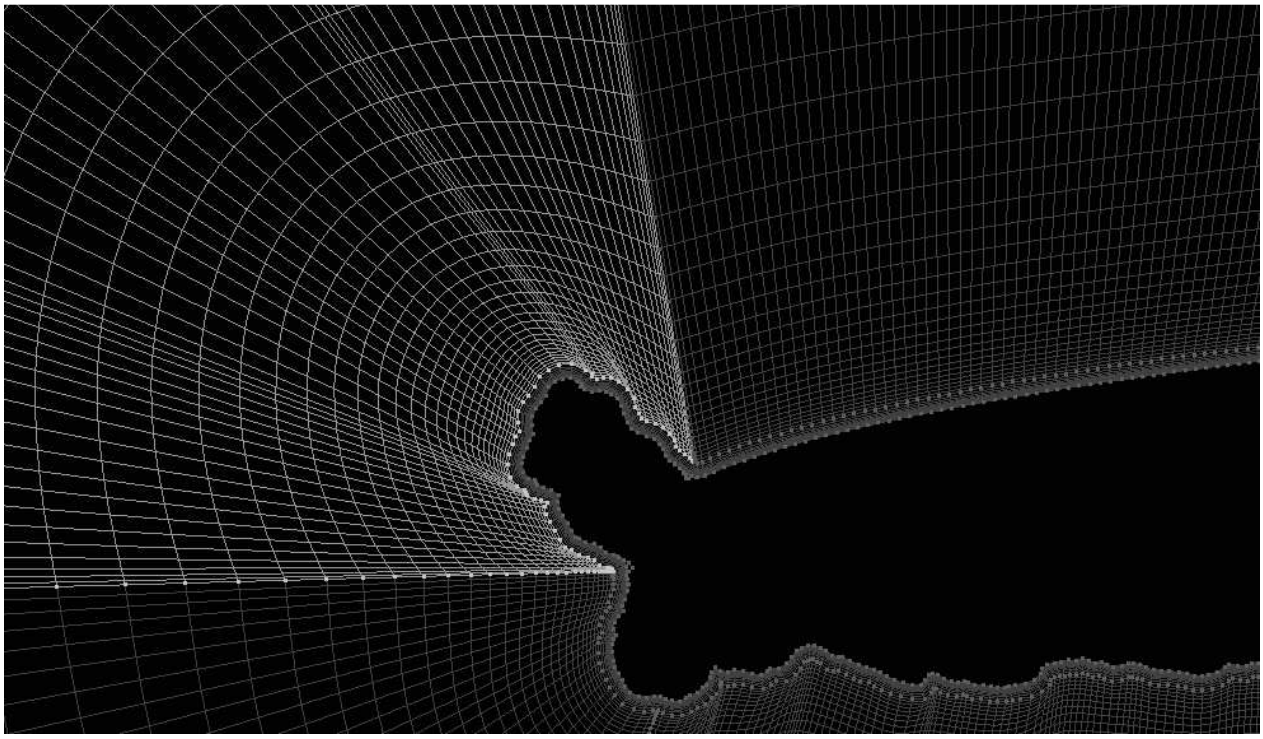

Figure 8. Modify block boundary (step 1) – curve is selected.

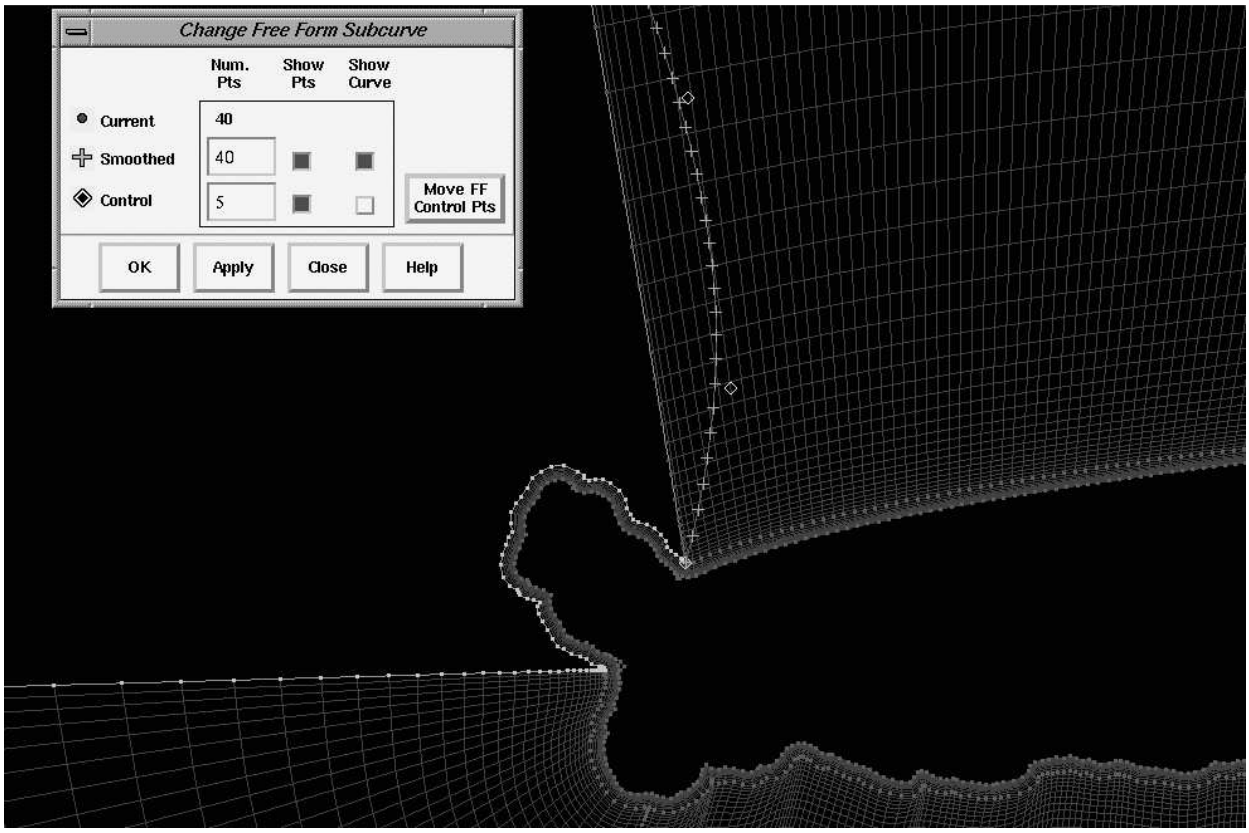

Figure 9. Modify block boundary (step 2) – shape of the curve is changed.

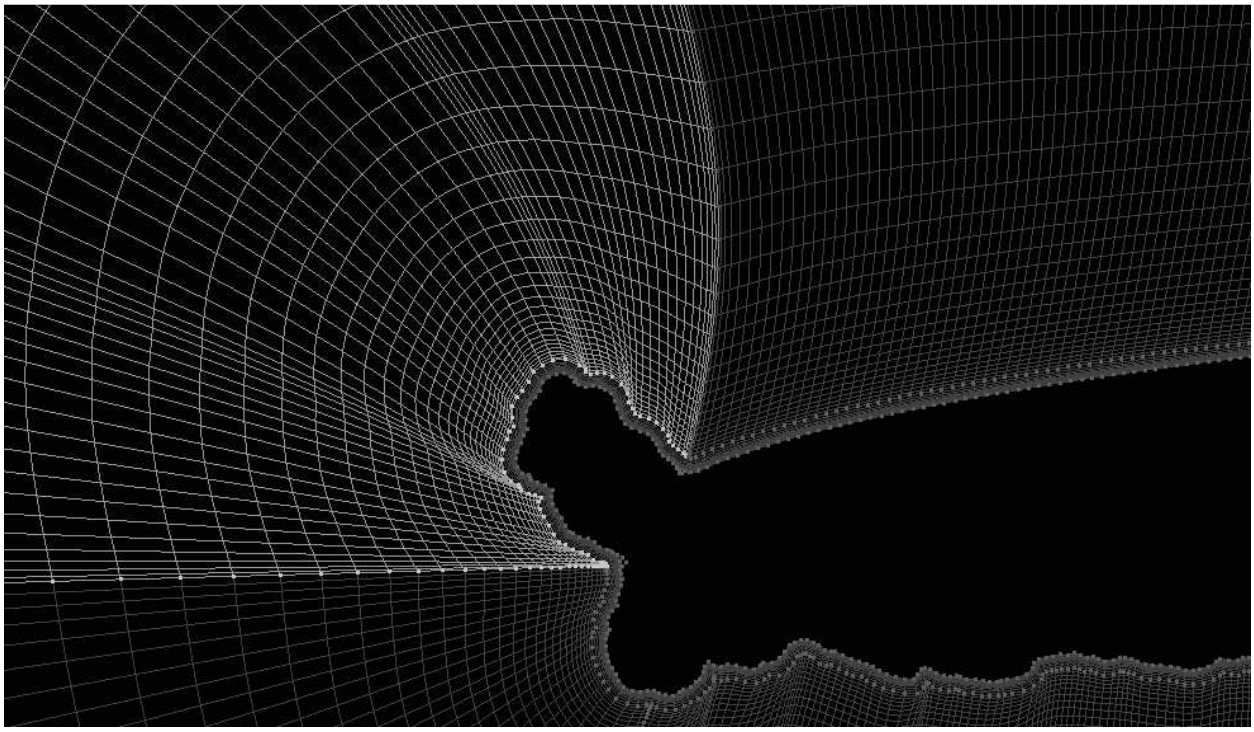

Figure 10. Modify block boundary (step 3) – grid is regenerated.

In this example, changes to a boundary are propagated to the shared edge of the one-to-one abutting block. Other methods of propagation may also be useful, and these will be provided by in SmaggIce v2.0.

8. Match cell spacing

Frequently, points need to be distributed along a block edge so that the cell spacings match those of the block's neighbors. SmaggIce provides a tool to do this. The spacing can match at either end or both ends of a block edge.

C. Grid generation and modification

To generate its grids, SmaggIce uses transfinite interpolation followed by elliptic smoothing. The elliptic smoothing is done using the ELLIP3D routine from a gridding package provided by David Saunders of NASA Ames Research Center. It uses Thomas Middlecoff background control functions. The transfinite interpolation is done using the TFI2D routine from the same package. Several parameters to the ELLIP2D function, such as number of iterations and tolerance, can be modified by the user.

SmaggIce can do two kinds of grid smoothing. The first again uses ELLIP2D to do elliptic smoothing on the grid of the currently selected block. The second kind of smoothing smoothes across two abutting blocks. The user first selects a subcurve on the shared boundary between the blocks. A sub-grid consisting of grid cells from both blocks on either side of the subcurve is then smoothed using Laplace smoothing. This smoothing necessarily modifies the location of the selected subcurve.

Other tools which allow the user to modify grids include dividing a gridded block, merging two gridded blocks which are abutting one-to-one, and copying the spacing of points along one edge of a gridded block to the interior grid lines.

D. Grid quality analysis

Grid quality measurements such as aspect ratio, orthogonality, skewness, and stretching can be calculated and displayed graphically on the gridded blocks as color-coded overlays. Minimum and maximum values for these measurements are also displayed. Figure 11 shows grid cells colored by orthogonality. This allows the user to quickly identify areas of poor grid quality, which may then be repaired.

E. Output

There are several options for output in SmaggIce. 2D grids can be saved in PLOT3D format. Airfoil geometry can also be extruded into 3D geometry by applying sinusoidal variations in the spanwise direction with a specified

sweep angle. This 3D geometry can be saved in PLOT3D or point cloud format (which simply lists the x,y,z coordinates of each point of the surface).

A goal of the SmaggIce project is to use CGNS[17] as a file format to transfer information to and from other CFD applications like post-processors and flow solvers like WIND. Using the CGNS format will allow ice geometry, grids, boundary conditions, connectivity, and solutions to all be saved in a single file. While we have had some success with transferring data from SmaggIce to post processors (like TecPlot) using CGNS, we have been unable to make SmaggIce-generated CGNS files work with WIND. We are working with the developers of CGNS and WIND to fix this problem. The CGD file format currently used by WIND is being considered as an alternative in case the problem cannot be resolved.

F. General/convenience capabilities

1. Scripting

A graphical user interface (GUI) is provided to interact with the SmaggIce software. A scripting interface is also provided to the user, since it offers several benefits. Frequently done steps in a gridding process can be saved to a script file and run over and over again. This saves time and reduces user error compared to repeating these steps using the GUI. Scripting enables the recording feature of SmaggIce. By turning on recording, all of the user's actions are recorded to a script file which can then be replayed. The file can also serve as documentation of the steps that the user took to get a certain state in SmaggIce. Since scripts are easy-to-read text files, they can be edited and reused for parametric studies. An example script is shown in Figure 12.

The scripts are written in the Python programming language. Python is powerful programming language with a very clear syntax that is frequently used as a scripting language for engineering and research applications.[18] SmaggIce has extended Python to include functions specific to SmaggIce.

If SmaggIce encounters an error while trying to run a script, a dialog window lets the user know that an error occurred and that more information about the error is given in the Message window. All SmaggIce scripting functions throw an exception if there is an error so the script writer can use standard Python exception handling methods in the script.

```
FileRead( '/usr/local/u1/geometry/944.exp' , 'ELEMENT' );
SetRefAirfoil();
SmgChooseSubcurve( 561, 38 );
SmgRediscretizeSubcurve( 250 , 4 , 0 );
SmgWake( 40 , 15 , 3 );
SmgViscSublayer( 20 , 0.003 , 1e-05 , SMG_true );
SmgNearFieldRadialCuts( 0.6 , 20 , [ ( 78, SMG_lowerLeg, 0.393983 ),
( 232, SMG_radius, 47.3174 ), ( 268, SMG_radius, 90 ), ( 324, SMG_radius, 157.554 ),
( 378, SMG_upperLeg, 0.268703 ) ] );
SmgSelectObject(6);
SmgChooseSubcurve( 0, 19, EDGE_IMAX );
SmgChangeFreeForm( 18 , 4 , [ ( 3, -0.0717987, 0.155125 ) ,
( 2, 0.0240446, 0.024011 ) ]);
SmgTanhRedist( 0 , 61 );
SmgGenGrid( 3 );
SmgOuterblock( 3 , 2 , 150 , 20 , 15 );
```

Figure 12. Sample script file.

2. Undo/Redo

It is extremely important that a complex, graphically oriented interactive program like SmaggIce have undo capability. SmaggIce has full undo and redo capability for all user actions that affect the data defining the elements, wake, and blocks. The maximum number of undo levels is set by the user via a preference window.

3. Save state

The save state feature provides a mechanism to save the "state" of the SmaggIce application to a file. The state consists of all information about the elements, blocks, grids, and settings from the current session. Restoring the state consists of reading the state file into SmaggIce. Saving the state before quitting the program allows the user to restart SmaggIce and quickly recover the previous session, without having to redo all the steps from a previous session.

4. Preferences

Preferences are used to set the default behavior for certain operations. For example, gridding preferences include tolerance, the maximum number of iterations, and relaxation factors. Preferences can be set using a GUI window and will be used for the current session. They may also be saved to a file for use in all subsequent sessions. The preference file is a text file which can be edited by the user, if desired.

5. Help

Updated help has been added for new features. The help files are available as HTML and can be viewed using a Web browser (e.g., Netscape or Internet Explorer). They may be displayed from within SmaggIce or can be viewed from outside the program.

6. Portability

SmaggIce is intended to run on UNIX and MS-Windows platforms. It has been tested on SGI (IRIX), Sun (Solaris), Intel (Red Hat Linux), and several MS-Windows platforms.

To run SmaggIce on a UNIX platform, you should have Web browser software for viewing help files. If you are using the SmaggIce package that uses OpenGL, you must have the OpenGL libraries installed on your system.

To run SmaggIce on a PC running Microsoft Windows, you should have Microsoft Internet Explorer or Netscape Web browser software for viewing help files, and third party X-server software installed on your PC that allows you to display X Window applications. X-server software is available from various sources. SmaggIce has been tested with several X-servers (e.g., Hummingbird Exceed, HOBLinkX11, MI/X 4.0, Wina/XE, X-ThinPro).

G. Development details

SmaggIce is written in C and FORTRAN. C is used for the GUI, control, interaction, graphics, memory management, and other computational routines; FORTRAN is used for some computational routines.

SmaggIce contains an embedded Python interpreter. Python has been extended to let the user modify SmaggIce data using Python scripts in script files. For more information about Python see http://www.python.org.

The GUI was developed for the X Window System using Motif, Xt Intrinsics, and Xlib functions. This will aid in the portability of the user interface across multiple computer platforms running X. Whenever possible, the GUI uses higher level libraries (Motif widgets) rather than the functionally equivalent Xt or Xlib libraries, because the higher-level code hides many of the details. This makes the code less complex, so the application is more easily maintained.

OpenGL is used for the graphics drawing. It uses the GLX extensions to X to interface with the windowing system, but the Mesa library can be used if the client workstation does not have OpenGL or if the X server does not support the GLX extensions.

Dynamic memory management is used to allocate only as much memory as is necessary for data storage and access. This allows the program to process models whose size is limited only by the amount of memory on the host computer. It also allows multiple input files to be read in and processed during a single session. As objects are read in or created, memory is allocated for them to store the object type as well as attributes describing the object. Space is also allocated for the data points defining the geometry, connectivity, and boundary condition data, and pointers to those are stored with the object. When additional objects are read in or created, or as geometry is modified (e.g., points are added), memory is reallocated as needed. When objects are deleted, the memory is freed to make room for new objects.

IV. Plans for Version 2.0

Version 2.0 of SmaggIce is currently under development. Plans are to include tools to display solutions computed by CFD software and to provide more tools for modifying blocks and grids. After a solution is generated through the CFD flow solver (such as WIND), solution data (such as pressure or velocity vectors) will be able to be displayed as colored overlays on the gridded blocks. This will help the user identify areas where the grids may need to be modified. New block and grid modification tools will include grid stretching and refining, generalized divide block (not restricted to dividing along a grid line), block merging, and introducing abutting mismatched block edges, rather than requiring abutting one-to-one edges as in version 1.8.

Another enhancement will be the capability to easily change block boundaries and have those changes propagate throughout the grids (to shared edges and edges on the opposite side of the block). After the number of boundary points and/or their locations are modified along an edge of a block, it is often desirable to propagate those modifications to other block edges. Algorithms are being developed to allow automatic propagation of the number of points from the newly modified "donor" edge of the current block to the opposite "receiver" edge of that block, thus making the block grid-ready (i.e., the opposite edges of the block have the same number of points). Those al-

gorithms can preserve the approximate distribution (that is, the varying density of boundary points along the edge) of the old receiver points using linear interpolation or an efficient custom-developed smooth interpolation method. Alternatively, those algorithms can locate the new receiver points along the old receiver edge according to the distribution of the new donor points. Also, algorithms are being developed to allow automatic propagation of point distributions from the donor edge of the current block to shared receiver edges of adjacent blocks, either by forcing one-to-one connectivity or else by moving the old receiver points to the shape of the donor's new edge while preserving the old number and varying densities of receiver points and allowing abutting mismatched connectivity between the donor and receiver edges. Optionally, the user will be able to automatically apply those algorithms recursively outward from the donor edge of a near field block that was just modified, propagating the modifications to multiple near field block edges away from the iced airfoil until the outer block is reached.

V. Conclusion

This paper presents the current status and plans of ongoing development of a software toolkit, SmaggIce, that will streamline the entire process of icing aero analysis, from geometry to flow solution. The main focus was on grid generation tools which will be available in version 1.8, scheduled for release in February 2005. It contains several unique grid generation features for iced airfoils. To provide a complete picture of a fully developed SmaggIce 2D software system, a brief overview is given of geometry modeling in the previously-released version 1.2 and a discussion of plans for the final phase of the software in version 2.0 of SmaggIce.

SmaggIce version 1.8 will provide essential tools for grid generation over iced airfoils. Additional interactive and automatic features for domain decomposition and grid generation over iced airfoils will be implemented in version 2.0 to provide further efficiency and convenience. It should also be noted that SmaggIce v1.8 can also be used for grid generation by those who are interested in airfoil analysis and design without ice accretion.

References

1. Potter, M.C., Wiggert, D.C., Hondzo, M., and Shih, T.I-P, "Mechanics of Fluids," 3rd edition, Brooks/Cole, Pacific Grove, California, 2001.
2. Choo, Y.K, Vickerman, M.B., Hackenberg, A.W., Rigby, D.L., "An Aerodynamic Simulation Process for Iced Lifting Surfaces and Associated Issues," SAE Paper 2003-01-2135, June 2003.
3. Thompson, D., Mogili, P., Chalasani, S., Addy, H., Choo, Y., "A Computational Icing Effects Study for a Three-Dimensional Wing: Comparison with Experimental Data and Investigation of Spanwise Variation," AIAA Paper 2004-0561, Jan. 2004.
4. Mogili P., Thompson, D. S., Choo, Y., and Addy, H., "RANS and DES Computations for a Wing with Ice Accretion," AIAA Paper 2005-1372, January 2005.
5. Chi, X., Zhu, B., Addy, H.E., Choo, Y.K., and Shih, T.I-P., "A Comparative Study Using CFD Techniques and Turbulence Models to Predict Iced Airfoil Aerodynamics," AIAA Paper 2005-1371, Aerospace Sciences Meeting, Reno, Nevada, January 2005.
6. Communication with Tom I-P Shih and Rich Hindman of Iowa State University, August 2005.
7. DeYoung and Harpter, NACA Report 921, 1948.
8. Weissinger, NACA TM 1120, 1947.
9. Choo, Y.K., Slater, J.W., Vickerman, M.B., Van Zante, J.F., "Geometry Modeling and Grid Generation for Computational Aerodynamic Simulations around Iced Airfoils and Wings," Proceedings of 8th International Conference on Numerical Grid Generation in Computational Field Simulations, edited by Soni et al., pp. 561-570, June 2002.
10. Vickerman, M.B., Choo, Y.K., Schilling, H.S., Baez, M., Braun, D.C., Cotton, B.J., "Toward an Efficient Icing CFD Process Using an Interactive Software Toolkit – SmaggIce 2D", AIAA Paper 2002-0380, January 2002.
11. Bush, R.H., Power, G.D., Towne, C.E., "WIND: The Production Flow Solver of the NPARC Alliance", AIAA Paper 98-0935, January 1998.
12. Cotton, B.J., Baez, M., Vickerman, M.B., "SmaggIce Users Guide, Version 1.2", Feb. 2003.

13. Chung, J., Reehorst, A., Choo, Y., and Potapczuk, M., "Effects of Airfoil Ice Shape Smoothing on the Aerodynamic Performances," AIAA Paper 98-3242 AIAA/ASME/SAE/ASEE Joint Propulsion Conference, Cleveland, July, 1998.

14. Addy, H.E., "Ice Accretions and Icing Effects for Modern Airfoils," NASA/TP-2000-210031/SUPPL, April 2000.

15. Broeren, A., Addy, H., and Bragg, M., "Flowfield Measurements About an Airfoil with Leading Edge Ice Shapes", AIAA Paper 2004-0559, Reno, NV, Jan. 2004.

16. Bragg, M. B., Broeren, A. P., and Blumenthal, L. A., "Iced Airfoil and Wing Aerodynamics", ASE Paper 2003-01-2098, June 2003.

17. Legensky, S.M., Edwards, D.E., Bush, R.H., Poirier, D.M.A., Rumsey, C.L., Cosner, R.R., Towne, C.E., "CFD General Notation System (CGNS): Status and Future Directions", AIAA Paper 2002-0752, January 2002.

18. Python documentation, http://www.python.org/doc, May 27, 2004.